

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП'ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА СИСТЕМ ШТУЧНОГО ІНТЕЛЕКТУ



Лабораторна робота №1

на тему: *Знайомство з С. Виконання програми простої структури*

Виконав:

студент групи КН-109

Гречух Тарас

Прийняв:

Гасько Р.Т.

ЛЬВІВ 2018

Лабораторна робота №1

Тема роботи: *Знайомство з С. Виконання програми простої структури*

Мета роботи: Знайомство з середовищем програмування, створення, відлагодження й виконання простої програми, що містить ввід/вивід інформації й найпростіші обчислення.

Постановка завдання №1:

Обчислити значення виразу при різних дійсних типах даних (float й double). Обчислення варто виконувати з використанням проміжних змінних. Порівняти й пояснити отримані результати.

Код програми:

- З використанням *double*:

```
#include <stdio.h>

#include <math.h>

int main()
{
    double a,b,result;

    printf("Please input a: \n");
    scanf("%lf", &a);

    printf("Please input b: \n");
    scanf("%lf", &b);

    result = (pow((a-b),3)-(pow(a,3)-3*(a*a)*b))/(3*a*(b*b));

    printf("Result =%lf", result);

    printf("\n");

    return 0;
}
```

- З використанням *float*:

```
#include <stdio.h>

#include <math.h>

int main()
{
    float a,b,result;

    printf("Please input a: \n");

    scanf("%f", &a);

    printf("Please input b: \n");

    scanf("%f", &b);

    result = (pow((a-b),3)-(pow(a,3)-3*(a*a)*b))/(3*a*(b*b));

    printf("Result =%f", result);

    printf("\n");

    return 0;
}
```

Результат виконання програми

1. Результати роботи програми типу *double*:



```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 {
6     double a,b,result;
7
8     printf("Please input a: \n");
9     scanf("%lf", &a);
10
11     printf("Please input b: \n");
12     scanf("%lf", &b);
13
14     result = (pow((a-b),3)-(pow(a,3)-3*(a*a)*b))/(3*a*(b*b));
15
16     printf("Result =%lf", result);
17     printf("\n");
18
19 [harvard@appliance ~]: ./lab11float
Please input a:
1000
Please input b:
0.0001
Result = 2207030.000000
[harvard@appliance ~]: make lab11
make: 'lab11' is up to date.
[harvard@appliance ~]: ./lab11
Please input a:
1000
Please input b:
0.0001
Result = 1.001350
[harvard@appliance ~]:
```

2. Результати роботи програми типу *float*:



```
1#include <stdio.h>
2#include <math.h>
3
4int main()
5{
6    float a,b,result;
7
8    printf("Please input a: \n");
9    scanf("%f", &a);
10
11    printf("Please input b: \n");
12    scanf("%f", &b);
13
14    result = (pow((a-b),3)-(pow(a,3)-3*(a*a)*b))/(3*a*(b*b));
15
16    printf("Result =%f", result);
17    printf("\n");
18}
jharvard@appliance (~): ./lab1float
Please input a:
1000
Please input b:
0.0001
Result =1.001358
jharvard@appliance (~): make lab1float
make: 'lab1float' is up to date.
jharvard@appliance (~): ./lab1float
Please input a:
1000
Please input b:
-2267030.000000
Result =-2267030.000000
jharvard@appliance (~):
```

Пояснення результатів:

Тип double обраховує з більшою точністю ніж float, та має більший діапазон значень, але займає у оперативній пам'яті 8(64) байти(біти) , а float – 4(32) байти.

Постановка завдання №2:

Обчислити значення виразів. Пояснити отримані результати.

Код програми:

```
#include <stdio.h>
#include <math.h>
```

```
int main()
{
    int n,m,r1,r2,r3;
```

```
    printf("Input n: \n");
    scanf("%i", &n);
    printf("Input m: \n");
    scanf("%i", &m);
```

```
    r1 = --m - ++n;
    r2 = (m < n++);
    r3 = (n-- > m++);
```

```
    printf("Result 1=%i \n", r1);
    printf("Result 2=%i \n", r2);
```

```
printf("Result 3=%i \n", r3);

return 0;
}
```

Результати програми:

При n=12, m=8:



```
make: lab1 is up to date.
jharvard@appliance (~): ./lab1
Input n:
12
Input m:
8
Result 1=-6
Result 2=1
Result 3=1
jharvard@appliance (~):
```

При n=13, m=21:



```
jharvard@appliance (~): ./lab1
Input n:
13
Input m:
21
Result 1=0
Result 2=0
Result 3=0
jharvard@appliance (~):
```

Пояснення результатів:

n++ та ++n це інкременти, а -- n та n-- це декремент. Якщо дія стоїть перед змінною, то вона спершу додає або віднімає від числа 1 і надалі число $n=n+1$ або $n=n-1$. Якщо дія стоїть після змінної, то вона в даній функції значення n не міняє, а додає 1 вже по закінченню даної, тобто в даній функції $n=n$ (так і залишається), а в наступній уже $n=n+1$. Відповідно так і обраховує змінні програма і виводить нам різницю двох змінних з інкрементом та декрементом та два твердження (True/False, 0/1) щодо порівняння змінних.

Прогрес в CS50:

На даний момент (20.09) я знаходжусь на тижні 2 і прослуховую лекцію 2-1.